Simple SQL & Relational Queries

[1] Query to Retrieve the Number of Orders for a Specific Customer :-

❖SQL Query :-

- SELECT c.id, COUNT(*) AS order_count FROM supply_chain.orders o, supply_chain.customer c WHERE o.customer_id = c.id And c.id = 1

  GROUP BY c.id;

❖Relational Query :-

- π customer_id, count(orders) (σ_customer_id=1 (orders ⋈ customer))

[2] Query to Retrieve all products that have the highest unit price :-

❖SQL Query :-

- SELECT * from supply_chain.products where unit_price in (select MAX(unit_price) from supply_chain.products);

❖Relational Query :-

- π (σ_unit_price=π_max(unit_price) (products))

[3] Query to Retrieve Supplier with total Products :-

❖SQL Query :-

- SELECT s.name AS supplier_name, COUNT(p.id) AS product_count FROM supply_chain.suppliers s , supply_chain.products p where s.id = p.supplier_id GROUP BY s.name;

❖Relational Query :-

- π_{s.name,ρ_{s.id,COUNT(p.id)→product_count}(⋈_{s.id=p.supplier_id}(π_{id, name}(suppliers) ⋈ π_{id}(products))))

[4] Query to Retrieve Shipped Shipments Sorted by Shipment Date :-

❖SQL Query :-

- SELECT * FROM supply_chain.shipments WHERE status = 'Delivered' ORDER BY shipment_date ASC;

❖Relational Query :-

- π shipment_id, transportation_mode, shipment_date, delivery_date, status(σ status=’Delivered’ (σ shipment_date ASC (shipments)))

[5] Count the number of shipments in transit:-

❖SQL Query :-

- SELECT COUNT(*) AS transit_shipments FROM supply_chain.shipments WHERE status = 'In Transit’;

❖Relational Query :-

- π_sum(quantity) (σ_status='In Transit' (orders ⋈ shipments))

Complex SQL & Relational Queries

[1] Query to Find Customers with Multiple Orders :-

❖SQL Query :-

- SELECT c.id AS customer_id, c.name AS customer_name, COUNT(o.id) AS order_count FROM supply_chain.customer c

  JOIN supply_chain.orders o ON c.id = o.customer_id

  GROUP BY c.id, c.name HAVING COUNT(o.id) > 1;

❖Relational Query :-

- π customer_id, customer_name, order_count (σ order_count>1 (γ customer_id, customer_name, order_count:COUNT() (customer⋈ id = customer_id orders)))

[2] Query to Retrieve Products Ordered by a Specific Customer with Shipment Details :-

❖SQL Query :-

- SELECT c.name AS customer_name, s.id AS shipment_id,s.status,s.delivery_date, w.address AS warehouse_address

  FROM supply_chain.customer c

  JOIN supply_chain.shipments s ON c.id = s.customer_id

  JOIN supply_chain.bridge b ON s.id = b.shipment_id

  JOIN supply_chain.warehouse w ON b.track_id = w.id

  WHERE c.id = 1;

❖Relational Query :-

- π customer_id, customer_name, order_count (σ order_count>1 (γ customer_id, customer_name, order_count:COUNT() (customer⋈ id = customer_id orders)))

[3] Query to Retrieve Products with the Lowest Total Order Quantity:-

❖SQL Query :-

- SELECT p.id AS product_id,p.name AS product_name, SUM(o.quantity) AS total_order

  FROM supply_chain.products p, supply_chain.suppliers s,supply_chain.orders o

  WHERE p.id = s.id AND s.id=o.supplier_id

  GROUP BY p.id

  ORDER BY total_order

  LIMIT 1;

❖Relational Query :-

- π product_id, product_name, total_order(γ product_id, product_name, total_order:SUM(quantity)(σp.id = s.id AND s.id = o.supplier_id(products×suppliers×orders)))

[4] Query to Find the Customers Who Have Not Placed Orders:-

❖SQL Query :-

- SELECT c.id AS customer_id, c.name AS customer_name
  FROM supply_chain.customer c
  LEFT JOIN supply_chain.orders o ON c.id = o.customer_id
  WHERE o.id IS NULL;

❖Relational Query :-

- π customer_id, customer_name(σ o.id IS NULL (customer$\ltimes$orders))

[5] Retrieve the Top 5 Customers with the Highest Total Order Value :-

❖SQL Query :-

- SELECT c.id AS customer_id,c.name AS customer_name, SUM(o.quantity * p.unit_price) AS total_order_value FROM supply_chain.customer c

  JOIN supply_chain.orders o ON c.id = o.customer_id

  JOIN supply_chain.suppliers s ON s.id = o.supplier_id

  JOIN supply_chain.products p ON p.id = s.id

  GROUP BY c.id, c.name

  ORDER BY total_order_value DESC

  LIMIT 5;

❖Relational Query :-

- πcustomer_id, customer_name, total_order_value(γ customer_id, customer_name, total_order_value:SUM(quantity \timesunit_price) (customer⋈ id = customer_id orders⋈ supplier_id = id suppliers⋈ id = supplier_id products))